N.E. Pages 58. Replace existing claim 1 with revised claim 1.

N.E. Pages 65-66. Replace existing claim 17 with revised claim 17.

N.E. Page 73. Replace existing claim 33 with the revised claim 33.

General Remarks

The applicant has corrected several confusing terminology clerical errors, and has corrected several forward references in the specification text.

In addition, the applicant has rewritten all claims to define the invention more distinctly and patentably over prior art that might be cited during prosecution.

Two physical pages have been provided for each page of the original specification that is being modified. One page shows the original, with markups. The second page contains revised text. Complete pages have been provided to make it easy to amend the original documents.

Conclusion

The applicant submits that the specification and claims are now in proper form, and that the revised claims define the invention patentably over the prior art.

Conditional Request For Constructive Assistance

The applicant is a first-time pro se inventor. The applicant has amended the specification and claims so that they are proper, definite, and define novel structure that is also unobvious. If the Examiner believes this application is not in full condition for allowance, the applicant respectfully requests the constructive assistance of the Examiner in accordance with MPEP 2173.02 and 707.07, in order that the applicant can place this application in allowable condition as soon as possible and without the need for further proceedings.

Respectfully yours,

Kevin W Jameson (Applicant Pro Se)

Patent Application for

Kevin W Jameson

Collection Adaptive Focus GUI

Cross References To Related Applications

The present invention uses inventions from the following patent applications, which are incorporated herein by reference:

Collection Information Manager USPTO Patent Application 09/885078 filed June 21, 2001, Kevin W Jameson.

Collection Knowledge System USPTO Patent Application 09/885079 filed June 21, 2001, Kevin W Jameson.

USPTO 10/003,421 filed December 6 2001

Collection Extensible Action GUI, USPTO Patent Application filed contemporaneously herewith, Kevin W Jameson.

USPTO 10/003,423 filed December 6, 2001

Collection Role Changing GUI, USPTO Patent Application filed contemporaneously herewith, Kevin Jameson.

Field of the Invention

This invention relates to graphical user interfaces for processing collections of computer files in arbitrary ways, thereby improving the productivity of software developers, web media developers, and other humans that work with collections of computer files.

214 Module set situation focus variable groups

250 Module Redisplay GUI layout

Detailed Description

Overview of Collections

This section introduces collections and some related terminology.

Collections are sets of computer files that can be manipulated as a set, rather than as individual files. Collection are comprised of three major parts: (1) a collection specifier that contains information about a collection instance, (2) a collection type definition that contains information about how to process all collections of a particular type, and (3) optional collection content in the form of arbitrary computer files that belong to a collection.

Collection specifiers contain information about a collection instance. For example, collection specifiers may define such things as the collection type, a text summary description of the collection, collection content members, derivable output products, collection processing information such as process parallelism limits, special collection processing steps, and program option overrides for programs that manipulate collections. Collection specifiers are typically implemented as simple key-value pairs in text files or database tables.

Collection type definitions are user-defined sets of attributes that can be shared among multiple collections. In practice, collection specifiers contain collection type indicators that reference detailed collection type definitions that are externally stored and shared among all collections of a particular type. Collection type definitions typically define such things as collection types, product types, file types, action types, administrative policy preferences, and other information that is useful to application programs for understanding and processing collections.

Collection content is the set of all files and directories that are members of the collection. By convention, all files and directories recursively located within an identified set of subtrees are usually considered to be collection members. In addition, collection specifiers can contain collection content directives that add further files to the collection membership. Collection content is also called collection membership.

Collection is a term that refers to the union of a collection specifier and a set of collection content.

Collection information is a term that refers to the union of collection specifier information, collection type definition information, and collection content information.

Collection membership information describes collection content.

Collection information managers are software modules that obtain and organize collection information from collection information stores into information-rich collection data structures that are used by application programs.

Collection Physical Representations -- Main Embodiment

Figures 1-3 show the physical form of a simple collection, as would be seen on a personal computer filesystem.

FIG 1 shows an example prior art filesystem folder from a typical personal computer filesystem.

FIG 2 shows the prior art folder of FIG 1, but with a portion of the folder converted into a collection 100 by the addition of a collection specifier file FIG 2 Line 5 named "cspec". In this example, the collection contents 103 of collection 100 are defined by two implicit policies of a preferred implementation.

First is a policy to specify that the root directory of a collection is a directory that contains

a collection specifier file. In this example, the root directory of a collection 100 is a directory named "c-myhomepage" FIG 2 Line 4, which in turn contains a collection specifier file 102 named "cspec" FIG 2 Line 5.

Second is a policy to specify that all files and directories in and below the root directory of a collection are part of the collection content. Therefore directory "s" FIG 2 Line 6, file "homepage.html" FIG 2 Line 7, and file "myphoto.jpg" FIG 2 Line 8 are part of collection content 103 for said collection 100.

FIG 3 shows an example physical representation of a collection specifier file 102, FIG 2 Line 5, such as would be used on a typical personal computer filesystem.

Collection Information Types

Figures 4-5 show three ~~main~~ kinds of information that are managed by collections.

FIG 4 shows a high-level logical structure of three types of information managed by collections: collection processing information 101, collection specifier information 102, and collection content information 103. A logical collection 100 is comprised of a collection specifier 102 and collection content 103 together. This diagram best illustrates the logical collection information relationships that exist within a preferred filesystem implementation of collections.

FIG 5 shows a more detailed logical structure of the same three types of information shown in FIG 4. There is only one instance of collection type information 101 per collection type. There is only one instance of collection content information per collection instance. Collection specifier information 102 has been partitioned into collection instance processing information 104, collection-type link information 105, and collection content link information 106. FIG 5 is intended to show several important types of information 104-106 that are contained within collection specifiers 102.

Suppose that an application program means 110 knows (a) how to obtain collection processing information 101, (b) how to obtain collection content information 103, and (c) how to relate the two with per-collection-instance information 102. It follows that

FIG 6 110

application program means 110 would have sufficient knowledg to use collection processing information 101 to process said collection content 103 in useful ways.

Collection specifiers 102 are useful because they enable all per-instance, non-collection-content information to be stored in one physical location. Collection content 103 is not included in collection specifiers because collection content 103 is often large and dispersed among many files.

All per-collection-instance information, including both collection specifier 102 and collection content 103, can be grouped into a single logical collection 100 for illustrative purposes.

Collection Application Architectures

Figures 6-7 show example collection-enabled application program architectures.

FIG 6 shows how a collection information manager means 111 acts as an interface between an application program means 110 and collection information means 107 that includes collection information sources 101-103. Collectively, collection information sources 101-103 are called a collection information means 107. A collection information manager means 111 represents the union of all communication mechanisms used directly or indirectly by an application program means 110 to interact with collection information sources 101-103.

FIG 7 shows a physical software embodiment of how an application program means 110 could use a collection information manager means 111 to obtain collection information from various collection information API (Application Programming Interface) means 112-114 connected to various collection information server means 115-117.

Collection type definition API means 112 provides access to collection type information available from collection type definition server means 115. Collection specifier API means 113 provides access to collection specifier information available from collection

Claims

Replace claim 1 with revised claim 1.

I claim:

1. A Collection Adaptive Focus GUI process for adapting a graphical user interface to a new work situation, comprising the following steps:

   (a) receiving a work situation change event, and

   (b) performing an adaptive response to said work situation change event,

   thereby providing a solution to the Adaptive Focus GUI Problem, and

   thereby providing graphical user interfaces with a practical means for adapting themselves to changes in user focus and work situations, in a way that was not previously available.

2. The process of claim 1, wherein

   (a) said step of receiving a work situation change event receives an event selected from the group consisting of initial invocation change events and full work situation change events and partial work situation change events and partial work situation context change events and partial work situation base directory change events and partial work situation collection change events and partial work situation role change events and partial work situation timeset change events and partial work situation focus variable change events and partial work situation focus variable group change events,

   thereby helping to solve the Adaptive Focus GUI Problem, and

Adaptation Problem, and

thereby providing users with one or more visual indications of a GUI focus
change from a previous work situation to said new work situation, and

thereby providing users with a new set of work operations that are relevant to
said new work situation.

16. The process of claim 1, wherein

(a) said step of performing an adaptive response communicates adaptation
results to one or more destinations selected from the group consisting of
computer memories and computer display screens and computer files and
computer networks,

thereby helping to solve the Collection Adaptive Focus GUI Problem,

and thereby providing a practical means for displaying and storing adaptation
results as part of said adaptive response.

Replace claim 17 with revised claim 17.

17. A programmable Collection Adaptive Focus GUI device for adapting a graphical
user interface to a new work situation, whose actions are directed by software
executing a process comprising the following steps:

(a) receiving a work situation change event, and

(b) performing an adaptive response to said work situation change event,

thereby providing a solution to the Adaptive Focus GUI Problem, and

thereby enabling graphical user interfaces to adapt themselves to changes in

user focus and work situations in a scalable way that was not previously
available. *Replace claim 17 with revised claim 17*

18. The programmable device of claim 17, wherein

(a) said step of receiving a work situation change event receives an event
selected from the group consisting of initial invocation change events and full
work situation change events and partial work situation change events and partial
work situation context change events and partial work situation base directory
change events and partial work situation collection change events and partial
work situation role change events and partial work situation time change events
and partial work situation focus variable change events and partial work situation
focus variable group change events,

thereby helping to solve the Adaptive Focus GUI Problem, and

thereby helping to solve the Work Purpose Adaptation Problem, the Work
Location Adaptation Problem, the Work Object Type Adaptation Problem, the
Work Role Adaptation Problem, the Work Time Adaptation Problem, the Work
Method Adaptation Problem, the Work Object Instance Adaptation Problem, and

thereby enabling graphical user interfaces to respond to events corresponding to
work situation change concepts of why, where, what, who, when, and how.

19. The programmable device of claim 17, wherein

(a) said step of receiving a work situation change event receives a work situation
change event from a source selected from the group consisting of human
operators and external programs and a GUI program that is executing said step
of receiving a work situation change event,

computer memories and computer display screens and computer files and computer networks,

thereby helping to solve the Collection Adaptive Focus GUI Problem,

and thereby providing a practical means for displaying and storing adaptation results as part of said adaptive response.

Replace claim 33 with revised claim 33.

33. A computer readable memory, encoded with data representing a Collection Adaptive Focus GUI program that can be used to direct a computer when used by the computer, comprising:

(a) means for receiving a work situation change event, and

(b) means for performing an adaptive response to said work situation change event,

thereby providing a solution to the Adaptive Focus GUI Problem, and

thereby enabling graphical user interfaces to adapt themselves to changes in user focus and work situations in a scalable way that was not previously available.

34. The computer readable memory of claim 33, wherein

(a) said means for receiving a work situation change event receives an event selected from the group consisting of initial invocation change events and full work situation change events and partial work situation change events and partial work situation context change events and partial work situation base directory change events and partial work situation collection change events and partial work situation role change events and partial work situation time change events